

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 10-260832

(43)Date of publication of application : 29.09.1998

(51)Int.Cl.

G06F 9/38

G06F 9/38

(21)Application number : 09-064327

(71)Applicant : HITACHI LTD

(22)Date of filing : 18.03.1997

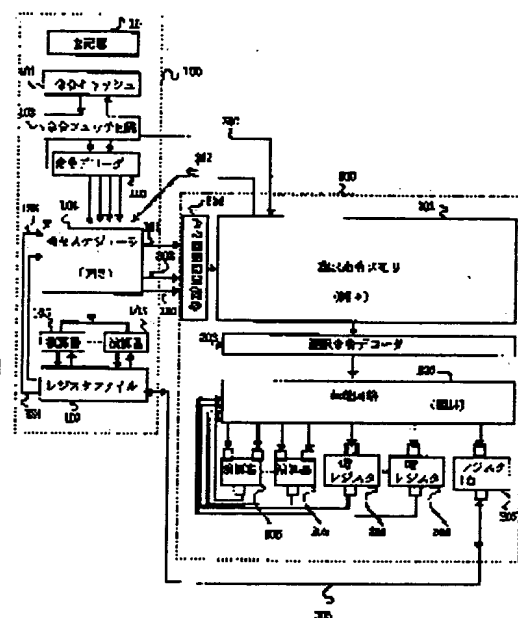
(72)Inventor : MIKI YOSHIO
TSUSHIMA YUJI

(54) INFORMATION PROCESSOR

(57)Abstract:

PROBLEM TO BE SOLVED: To shorten the schedule time to parallelly execute an instruction for successive execution.

SOLUTION: An instruction interpretation circuit 204 in a processor 200 generates a group of interpretation instructions to branch blocks of a program and stores it in interpretation instruction memory 201 when a scalar processor 100 carries out the program. Each interpretation instruction includes the numbers of physical resources (a computing element, a register, etc.) of a transfer source and a transfer destination which are decided by an instruction scheduler 104 to each of plural instructions that are decided as parallelly executable by the scheduler 104. When the branch block is executed again later, an interpretation instruction string in the memory 201 is successively carried out. A transfer circuit 203 supplies data to an input of a computing element that is selected for an instruction when the computing element of the transfer source generates the data that is used for an operation which is requested by each interpretation instruction.



【特許請求の範囲】

【請求項1】複数の第1種の演算器と複数の第1種のレジスタを含む複数の第1種の物理資源と、

実行すべきプログラムに含まれた複数の第1種の命令の内、並列に実行可能な複数の第1種の命令を選択し、それぞれの第1種の命令の実行に使用する複数の第1種の物理資源をそれらの選択された複数の第1種の命令に割り当て、それらの複数の第1種の命令のそれぞれに割り当てられた物理資源を使用してそれらの第1種の命令を並列に実行するように上記選択された複数の第1種の命令の実行を制御する命令スケジューラと、

上記命令スケジューラにより複数の第1種の命令が並列に実行されるごとに、該並列に実行された複数の第1種の命令の各々に対して上記命令スケジューラにより割り当てられた複数の物理資源を識別可能な少なくとも一つの第2種の命令を記憶するメモリと、

複数の第2種の演算器と複数の第2種のレジスタを含む複数の第2種の物理資源と、

上記プログラムが再度実行されたときに、そのプログラムに含まれた複数の第1種の命令に代えて、該複数の第1種の命令に対応して上記メモリに記憶された少なくとも一つの第2種の命令を実行する命令実行回路とを有し、

上記命令実行回路は、その第2種の命令に対応する上記複数の第1種の命令の各々に対して割り当てられた、上記一つの第2種の命令から識別可能な複数の物理資源に対応する複数の第2種の物理資源を使用してその第2種の命令を実行する情報処理装置。

【請求項2】上記複数の第2種の演算器は、上記複数の第1種の演算器とは別に設けられ、上記複数の第2種のレジスタは、上記複数の第1種のレジスタとは別に設けられている請求項1記載の情報処理装置。

【請求項3】上記複数の第1種の演算器が上記複数の第2種の演算器として使用され、上記複数の第1種のレジスタが上記複数の第2種のレジスタとして使用される請求項1記載の情報処理装置。

【請求項4】上記複数の第2種の演算器は、上記複数の第1種の演算器と同じ演算器を含み、上記複数の第2種のレジスタの数は、上記複数の第1種のレジスタの数と等しい請求項1記載の情報処理装置。

【請求項5】上記複数の第2種の演算器は、上記複数の第1種の演算器と異なる数の演算器を含み、上記複数の第2種のレジスタの数は、上記複数の第1種のレジスタの数とは異なる請求項1記載の情報処理装置。

【請求項6】複数の演算器と複数のレジスタを含む複数の物理資源と、

それぞれ一組の並列に実行可能な複数の第1種の命令に対応する複数の第2種の命令を記憶するメモリと、

上記翻訳メモリに記憶された複数の第2種の命令を順次実行する命令実行回路とを有し、

上記複数の第2種の命令は、それぞれの第2種の命令は対応する複数の第1種の命令が要求する演算を実行するために使用する複数の物理資源を識別可能に上記メモリに記憶され、

上記命令実行回路は、各第2種の命令に対応する複数の第1種の命令の各々に対して割り当てられた、上記第2種の命令から識別可能な複数の物理資源を使用してその第2種の命令を実行する情報処理装置。

【請求項7】複数の演算器と、

10 複数のレジスタと、命令によって使用される演算器およびレジスタを決定し、命令実行の入力となるレジスタ内データの更新を監視することによって、実行可能な状態となった命令を順次演算器へ投入する命令スケジューラと演算器へ投入された上記命令の命令語格納アドレスと使用される演算器およびレジスタ情報を記憶するメモリと、

既に上記メモリに格納されている命令語格納アドレスの命令を再度実行する際に、上記メモリに格納されている使用する演算器とレジスタに関する情報に基づきその命令を実行する命令実行回路とを有する情報処理装置。

【請求項8】複数の演算器と、

20 複数のレジスタと、命令によって使用される演算器およびレジスタを決定し、命令実行の入力となるレジスタ内データの更新とさらに命令実行の入力となるレジスタ内データを生成する演算器を監視することによって、命令実行に必要なデータの演算器からレジスタまたは演算器から演算器への転送経路を決定する転送回路と、演算器へ投入された上記命令の命令語格納アドレスと命令実行に必要なデータの転送経路情報を記憶するメモリとを有し、

30 既に上記メモリに格納されている命令語格納アドレスの命令を再度実行する際には、上記転送回路は、上記メモリに格納されている転送経路情報に基づき演算器とレジスタの間のデータ転送経路を切り替えて上記命令を実行する回路を有する情報処理装置。

【請求項9】複数の演算器と、

複数のレジスタと、

上記複数の演算器および複数のレジスタ間のデータ転送を可能とする出力線と入力線との間の複数の交点位置に設けられた、実行待ちの命令を待機させるための複数の命令待機回路と、

40 命令実行に必要なデータを生成する演算器または命令実行に必要なデータを保持しているレジスタと演算が実行される演算器との識別情報をデータ転送情報として持つ実行待ちの命令を、上記複数の命令待機回路の内、その命令の実行のためにデータ転送が必要となる演算器またはレジスタの入出力線の交点位置に設けられた一つの命令待機回路に投入する回路とを有する情報処理装置。

【発明の詳細な説明】

50 【0001】

【発明の属する技術の分野】本発明は、逐次実行用のプログラムの命令を並列に実行するスーパースカラー方式の情報処理装置に関する。

【0002】

【従来の技術】命令により制御される情報処理装置の処理速度を向上させるために、命令列が潜在的に有する並列実行性を利用する方法がある。例えば、マイク・ジョンソン著“スーパースカラー・プロセッサ”日経BP出版センタ、1994年、第104頁から第118頁および第125頁から第144頁（以下、文献1と呼ぶ）に示されているようなスーパースカラー方式では、複数の演算器が1つのプロセッサ内に設けられ、複数の命令を異なる演算器を使って並列に実行する。並列実行可能な命令は、命令実行ステージ以前のパイプラインステージ、例えば命令デコードステージにおいて、複数の命令が利用するハードウェア資源の競合や、命令実行に伴うデータの依存関係を調べることによって抽出される。

【0003】文献1の125ページ以降に記載されているように、アウトオブオーダー方式も可能である。この方法では、並列実行可能な命令を検索する範囲を広げ、並列実行可能な命令を元の命令順序とは異なる順序で実行する。

【0004】アウトオブオーダー実行を実現する方法として、文献1の109頁に記載されているTomasuloアルゴリズムが知られている。Tomasuloアルゴリズムでは、ある命令のデコードが終了し、その命令が利用する演算器や必要とする入力データが解読された後に、その命令がリザーベーションステーションと呼ばれる一種の命令バッファに格納される。リザーベーションステーションは実行待ちの命令格納場所として用いられ、演算器で実行終了した命令は、その終了をリザーベーションステーション内の全ての命令に告知する。これにより、待ち状態にある命令は自分自身の実行可能性を調べることができ、実行可能となった命令から順に演算器へ送られる。その結果として元の命令列順序とは異なった順序で命令を実行可能となる。2命令間の細かな依存関係の利用法としては、Hennesy and Patterson, "Computer Architecture A Quantitative Approach (second edition)", Morgan Kaufmann Publishers, Inc. 1995, PP. 147（以下、文献2と呼ぶ）に記載されているように、データフォワーディングが知られている。この方法では、或るレジスタの値を入力データとする命令が存在した時に、そのレジスタの値が確定した後に命令の実行を開始するのではなく、そのレジスタの値を更新しようとしている演算器から直接に値をその命令に転送し、レジスタの値が確定する前にその命令の実行を開始する。

【0005】このように、命令の実行に先立ってそれら

の命令間の関係を認識する方式は、命令が利用するハードウェア資源を具体的に決定すること、必要なデータが準備でき次第命令実行を開始することを制御方式として実現している。しかし、この制御方式を実現するための組み合わせ論理は大規模になる傾向が強く、ややもすれば高速実行のために設けた回路によって、全体の実行時間が消費されるという事態に陥る可能性もある。このような事態を回避するための一案として、VLIW (Very Long Instruction Word) (文献2、第278頁-第289頁)が提案されている。VLIWではコンパイラによって並列実行可能な複数の命令を予め抽出し、それらの命令を一つの命令語としてまとめておく。これにより、上記の認識に必要な回路規模を抑制することが可能となる。しかしながら、命令語が変わるために全てのプログラムをコンパイルし直す必要がある。

【0006】

【発明が解決しようとする課題】上述のように、VLIWは実行可能形式のバイナリーファイル資産を有効利用できない。さらに、スーパーコンピュータや信号処理専用プロセッサが対象とする、ループを多く使用する数値演算プログラムなどでは、コンパイルによるスケジューリングが可能であるが、これらはごく一部の特殊例に過ぎない。一般のプログラムでは、頻繁な分岐命令やメモリアクセスなど命令実行時間の動的変化が存在するため、コンパイラで静的にハードウェア利用順序を最適化するには限度がある。そこで、汎用プロセッサとしてはスーパースカラー方式のように、ハードウェアを用いた命令並列性の抽出および命令実行順序の決定、つまりスケジューリングが重要である。ところが、ハードウェアを用いたスケジューリングが必要とする論理量が膨大なことから、このスケジューリング部分の動作速度がプロセッサ全体の動作速度を律速してしまうという問題がある。

【0007】また、リザーベーションステーションでは、明らかにまだ実行不可能な命令に対しても、実行可能性のチェックが無駄に実施される。さらに演算器の数より多くの命令が実行可能となった場合には、実行すべき命令を選抜するなどの処理も必要である。この問題は、演算器における命令実行の前段階での時間浪費につながる。

【0008】本発明の目的は、より少ないオーバーヘッドで複数の命令を並列に実行できる情報処理装置を提供することにある。

【0009】本発明のより具体的な目的は、命令のスケジューリングによる命令の実行速度の低下を低減できる情報処理装置を提供することにある。

【0010】本発明の他のより具体的な目的は、命令が必要とするデータが準備でき次第、より少ない遅延でもってその命令の実行を開始できる情報処理装置を提供す

10

20

30

40

50

ることにある。

【0011】

【課題を解決するための手段】上記目的を達成するために、本発明による情報処理装置には、複数の第1種の演算器と複数の第1種のレジスタを含む複数の第1種の物理資源と、実行すべきプログラムに含まれた複数の第1種の命令の内、並列に実行可能な複数の第1種の命令を選択し、それぞれの第1種の命令の実行に使用する複数の第1種の物理資源をそれらの選択された複数の第1種の命令に割り当て、それらの複数の第1種の命令のそれぞれに割り当てられた物理資源を使用してそれらの第1種の命令を並列に実行するように上記選択された複数の第1種の命令の実行を制御する命令スケジューラとが設けられ、これらを用いてスーパースカラーモードで上記プログラムの第1種の命令が実行される。本発明では、上記命令スケジューラにより複数の第1種の命令が並列に実行されることに、該並列に実行された複数の第1種の命令の各々に対して上記命令スケジューラにより割り当てられた複数の物理資源を識別可能な少なくとも一つの第2種の命令を記憶するメモリと、複数の第2種の演算器と複数の第2種のレジスタを含む複数の第2種の物理資源と、上記プログラムが再度実行されたときに、そのプログラムに含まれた複数の第1種の命令に代えて、該複数の第1種の命令に対応して上記メモリに記憶された少なくとも一つの第2種の命令を実行する命令実行回路とが設けられ、上記命令実行回路は、その第2種の命令に対応する上記複数の第1種の命令の各々に対して割り当てられた、上記一つの第2種の命令から識別可能な複数の物理資源に対応する複数の第2種の物理資源を使用してその第2種の命令を実行する。

【0012】この結果、上記プログラム内の第1種の命令列を再度実行するときには、このメモリに記憶してある第2種の命令を実行することで、元の第1種の命令の実行時の命令の実行順序および使用する演算器等の物理資源の情報を利用することになる。この結果、命令スケジューラにより再度並列実行可能性の認識をする必要がなく、第2種の命令列はより高速に実行できる。

【0013】さらには、上記メモリおよび命令実行回路の動作クロックを、上記命令スケジューラのそれよりも高めることにより、第2種の命令列の実行速度を速めることも可能である。このためには、上記複数の第2種の物理資源は、上記複数の第1種の物理資源とは別に設けることが望ましい。しかし、これらの物理資源を共通の物理資源により実現することも可能である。

【0014】本発明のより具体的な態様では、各第2種の命令は、使用する物理資源と命令実行に必要なデータを生成する物理資源の情報を持ち、それらの情報を位置情報として対応づけられた第2種の命令の転送回路内に、各第2種の命令の実行を待ちあわせる命令待ち合わせ回路が設けられる。これにより、リザーベーションス

テーションで起きていた、明らかに実行が不可能な命令に対する実行可能性チェックや、実行直前での命令選抜に必要な論理や時間の削減が可能となる。

【0015】

【発明の実施の形態】以下、本発明に係る情報処理装置を、図面に示したいくつかの実施の形態を参照してさらに詳細に説明する。なお、以下においては、同じ参照番号は同じものもしくは類似のものを表すものとする。また、第2の実施の形態以降では、第1の実施の形態との相違点を主に説明する。

【0016】<発明の実施の形態>

(1)装置構成

図1において、情報処理装置は、二つのプロセッサ100と200により構成される。100は、逐次実行用に作成されたプログラムから命令の並列実行可能性を抽出し、個々のスカラ命令を並列に実行可能なスーパースカラープロセッサである。スーパースカラープロセッサ100は、主記憶10に接続された命令キャッシュ101と、命令フェッチ回路102、命令デコーダ103、命令スケジューラ104、複数の演算器105およびレジスタファイル107を有する。200は、スーパースカラープロセッサ100により実行されたプログラムが再度実行されるときに、そのプログラム内の命令をスーパースカラープロセッサ100よりも高速に実行するためのプロセッサである。プロセッサ200は、スーパースカラープロセッサ100より速いマシンクロックで動作するように構成されている。スーパースカラープロセッサ100において上記プログラムが最初に実行されたときに、命令スケジューラ104によるスケジューリングの結果として、並列に実行される命令のアドレス301、各命令に割り当てられた、演算器およびレジスタなどの物理資源の番号、具体的には、その命令の実行結果データを転送する転送先物理資源の番号302およびその命令の実行に使用するデータを供給する転送元物理資源番号303が供給される。このプロセッサ200では、命令翻訳回路204が、これらのスケジュール結果を反映した、それらの命令と等価な処理を指定する命令（以下これを翻訳命令と呼ぶ）を生成し、生成された翻訳命令列を翻訳命令メモリ201に記憶する。

【0017】プロセッサ200には、ここに記憶された翻訳命令を実行するための複数の演算器205と、これらの演算器が使用するデータあるいはこれらの演算器が生成した演算結果データを保持するための複数の一時レジスタ206と、これらの一時レジスタ206とレジスタファイル107との間でデータを交換するために使用されるレジスタ10207が設けられている。これらの演算器205の数は、スーパースカラープロセッサ100内の演算器105のそれと同じであり、一時レジスタ206の数もレジスタファイル107に含まれた、命令で指定可能なレジスタの数と同じである。もっとも、翻

訳命令で使用される物理資源数に依存してより少ない演算器205、一時レジスタ206で構成することも可能である。

【0018】翻訳命令メモリ201に記憶された翻訳命令の各々は、複数のスカラ命令に関する命令情報を含み、デコーダ202は、翻訳命令メモリ201から読み出された翻訳命令を解釈し、転送回路203は、デコーダ202により与えられる命令解釈情報が指定する複数の演算の各々が必要とするデータが上記複数の演算器205のいずれかから供給されるのに同期して、その演算を開始する。複数の演算器205は、複数のスカラ命令が要求する演算を並列に実行するとともに、スーパースカラープロセッサ100内の複数の演算器105よりも高速に演算を実行可能に構成されている。

【0019】転送回路203は、命令スケジューラ104と同じく実行すべき命令が実行可能になるのを待ち、実行可能になった時点でその命令を、いずれかの演算器に分配することを基本的な機能としている。しかし、転送回路203は、命令スケジューラ104と異なり、後述するように、どの先行命令の実行終了を待機すべきかについては転送経路中の位置として既にデコードされた状態となる。このために、転送回路203を用いた命令スケジューリングでは実質的に命令実行可能判定や実行順序を制御する回路が不要となり、命令スケジューラ104よりも高速処理を実行できる。

【0020】このように、プロセッサ200は、スーパースカラープロセッサ100内命令スケジューラ104のスケジュール結果を使用して、従って、このようなスケジュールを実行するよりも速いクロックで命令を実行する。なお、二つのプロセッサ100、200は必ずしも物理的に近接している必要性はないが、信号線301～305が通常のプロセッサ内部信号に匹敵する動作速度を必要とするため、例えば同一シリコン基板上に形成されるなどが好ましい。

【0021】(2) 命令の実行態様
スーパースカラープロセッサ100内の命令フェッチ回路102は、命令デコーダ103へ送出した命令のアドレスを命令アドレス300として翻訳命令記憶領域211にも供給する。図2を参照するに、プロセッサ200では、この命令に対する翻訳命令が翻訳命令メモリ201に記憶されているかを検索する(ステップ221)。もし、その命令アドレスに一致する命令ラインが翻訳命令メモリ201にあると(ステップ222)、そのラインに記憶された翻訳命令を読み出す(ステップ223)。翻訳命令デコーダ202は、読み出された翻訳命令をデコードし、その命令に含まれた転送元物理資源の情報を転送回路203内部の位置情報へ変換する(ステップ224)。転送回路203は実行に必要なデータが揃った命令から順に演算器205、一時レジスタ206、レジスタ10207へデータを転送する。データを

受け取った演算器205は演算結果を再び転送回路203へ戻し、演算命令が順次実行される(ステップ225)。ステップ222で翻訳命令記憶領域211内に命令アドレス300と同一の命令開始アドレスが発見されなかった場合には、プロセッサ200は作動せず、スーパースカラープロセッサ100による命令実行が実施される。すなわち、命令デコーダ103にて命令がデコードされ(226)、命令スケジューラ104で命令の実行順序が決定された(227)後、演算器105にて命令が実行される(228)。

【0022】(3) スーパースカラープロセッサ100
このプロセッサの概要は以下の通りである。命令キャッシュ101に格納された命令は、命令フェッチ回路102にてフェッチされ、命令デコーダ103に送り込まれる。命令デコーダ103は命令語が示す入出力レジスタの番号や演算種別の情報を解釈する。命令フェッチ回路102、命令デコーダ103は、いずれも複数の命令(一般的なスーパースカラープロセッサの能力として4～8命令)を同時に処理できるものとする。デコーダ103は、いわゆるレジスタリネーミング技法を用いて、プログラムで指定可能な論理的レジスタ番号をプロセッサ内部にある多量な物理レジスタ番号に変換する機能を持っていても良い。

【0023】命令スケジューラ104は命令デコーダ103から送られてきた複数の命令について、命令が利用する演算器の割り当てと命令実行順序の決定を行う。より具体的には、本実施の形態では、分岐命令処理ユニットも一つの演算器として扱われる。複数の演算器105には、整数演算器、浮動小数点演算器、分岐命令処理ユニットのように機能が異なるものおよび機能が同一の複数の演算器が混在している。演算器の個数はプロセッサで並列実行可能な命令数を制限するが、本実施の形態は、特定の数の演算器に限定はされない。いずれかの演算器105に対して複数の入力データラッチが存在する場合には、それぞれの入力データラッチに対して一意に識別可能な番号が付けられているものとする。この番号としては、例えばその入力データラッチに供給すべき演算結果データを出力する演算器の識別番号の下位ビット側に入力データラッチの番号を付加したものを利用できる。

【0024】命令の実行順序は命令実行に必要な入力データが揃った命令から順に実行することが原則となる。この制御には例えばTomastuloアルゴリズムを利用することができる。命令スケジューラ104には実行完了を待つべき先行命令が存在する命令が停留し、命令実行に必要な入力データが全て揃った命令は演算器105へ送出される。命令が完了するとレジスタの値が確定したという意味で値の確定したレジスタ番号がレジスタファイル107から命令スケジューラ104へ伝えられる。命令スケジューラ104は、停留している命令の中

に、伝えられた番号のレジスタを入力値として使用する命令があるか否か、入力値が全て揃った命令があるかを調べ、そのような命令があればその命令をいずれかの演算器105へ送出する。

【0025】以上説明した命令スケジューラ104の動作によって命令の実行順序が動的に決定され、その結果として各命令が使用した演算器、および命令の入力値となるデータを演算した演算器が決定できる。したがって、演算器やレジスタなどの物理資源を識別する番号を物理資源番号と定義すると、命令の動的な実行履歴はある物理資源からある物理資源へのデータ転送の履歴として見る事ができる。命令スケジューラ104は、演算器に送付した命令の命令アドレスである実行命令アドレス301、その命令によって起こるデータ転送の転送先と転送元をそれぞれ転送先物理資源302、転送元物理資源303などの命令実行履歴として命令翻訳回路204に送出する。

【0026】以下では、スーパーカラルプロセッサ100の回路の詳細をさらに説明する。命令フェッチ回路102は、フェッチした命令のアドレス300を命令デコーダ103とプロセッサ200に供給する。このようにプロセッサ200が実行すべき命令のアドレスを命令フェッチ回路102より取り出すと、プロセッサ200が実行すべき命令列をできるだけ早い時期に取り出すことになり、それ以降のスーパーカラルプロセッサ100がその命令に対して処理を実行するのを回避することができる。なお、実装面積等の制約により命令アドレス300を命令フェッチ回路102より取り出すことができない場合には、命令デコーダ103や命令スケジューラ104、演算器105といった命令実行ステージにより近い場所から命令アドレス300を取り出すことも可能である。命令フェッチ回路102は命令列の命令キャッシュ101からの命令の獲得を滞りなく実行するために、分岐命令の予測機能、命令バッファ機能等によって、実行可能性のある命令は先行的かつ余分に獲得するそれ自体公知の機能を有する。したがって、命令フェッチ回路102から命令デコーダ103へ送出された命令は情報処理装置で実行することが確定した命令列と考えることができる。

【0027】本実施の形態では図4に示したフォーマットの命令を用いる。この命令フォーマットはいわゆるRISCプロセッサで用いられる典型例であり、レジスタ番号等は特別な解釈なしに、命令のビットフィールドを分割するだけで得ることができる。Xフォーマットは一般の算術論理演算命令に用いられるフォーマットであり、冒頭の命令種を表すオペコード(OPCD)フィールド、演算結果が格納されるレジスタ番号であるターゲットレジスタフィールド(RT)、演算の入力数値が格納されるレジスタ番号として二つのオペランドレジスタフィールド(RA, RB)および拡張機能情報フィール

ド(EO)から構成される。Dフォーマットはロード、ストア命令のフォーマットである。オペコードフィールド、ターゲットレジスタフィールド、オペランドレジスタフィールドはXフォーマットと同様の機能を持ち、ディスプレースメントフィールド(D)はロード、ストア命令のアクセスするメモリ番地計算用の加算値を格納する。メモリ番地はオペランドレジスタ(RA)の値とディスプレースメントフィールドの値の加算値となる。オペコード(OPCD)フィールドから命令実行で利用される演算器の種類が判別でき、レジスタに関するRT, RA, RBフィールドから、命令が利用するレジスタが決定される。

【0028】図3において、スーパーカラルプロセッサ100内の命令デコーダ103は、命令フェッチ回路102から送られてきた命令を図4に例示した命令フォーマットにしたがって分割する。つまり、ラッチ110~113はそれぞれオペコード(opcd)、ターゲットレジスタ(RT)番号117、オペランドレジスタ(RA)番号118と、最後にオペランドレジスタ(RB)の番号またはディスプレースメント(D)の値を保持する。オペコードデコード回路115は、ラッチ110が保持するオペコードをデコードし、オペランドレジスタ(RB)を使用するか否かの判定信号114と命令で使用する演算器の物理リソース番号である演算器番号(FU)116とを生成する。シフト回路122は判定信号114がHighレベルのとき、命令フィールドの16-20ビットを下位27-31ビットにシフトし、上位16-20ビットは1を埋め直し、拡張データ(EX)119として出力する。判定信号114がLowのときには、命令フィールドの16-31ビットのディスプレースメント情報がそのまま拡張データ(EX)119として出力される。

【0029】命令スケジューラ104は、書き込みスコアボード120、読み出しスコアボード121および命令実行バッファ0~命令実行バッファ3を用いて先に述べた概念の命令スケジューリングを実行する。書き込みスコアボード120と読み出しスコアボード121はどちらもレジスタ番号をアドレス情報とするメモリである。書き込みスコアボード120はメモリデータとして各レジスタ番号毎に2ビットの領域を持ち、書き込もうとしているレジスタを読み出そうとしている先行命令が命令実行バッファ0から命令実行バッファ3までに幾つ存在するかを示すカウンタとなる。読み出しスコアボード121はメモリデータとして各レジスタ番号毎に1ビットの領域を持ち、演算器で実行中の命令の中に読み出そうとしているレジスタ内容を更新しようとする命令があるか否かを示す。命令実行バッファ0から命令実行バッファ3は先入れ先出し(FIFO)キューを構成しており、命令デコーダ103の出力である、演算器番号116、ターゲットレジスタ番号117、オペランドレジ

10

20

30

40

50

スタ番号118、拡張データ119の各信号と、命令フェッチ回路102からの命令アドレス125を保持する。命令選択回路123は、上記のFIFOキューに格納されている命令の内、実行可能で、最もFIFOキューに入ってから時間の経つものを選択する。

【0030】命令スケジューリングの詳細は次の通り、命令デコーダ103から送られてきた各レジスタ番号の情報は書き込みスコアボード120と読み出しスコアボード121とに入力される。その間に命令は命令実行バッファ0に取り込まれる。ただし、ヒット信号242がHighのときは翻訳命令が実行されるため、その間命令デコーダ103から送られてくる命令のスケジューリングおよび以降の動作はキャンセルされる。上記両スコアボードの出力値が0のときは、命令で使用するターゲットレジスタもオペランドレジスタも先行する命令によって使用中ではないことになる。つまり当該命令は実行可能であることがわかる。この段階で実行可能でない命令は、いずれかのレジスタの状態が変化するのを待つ必要があり、後続の命令が命令デコーダ103から到着するに従ってFIFOキューのより深い位置（命令実行バッファ1～3）へ進む。命令実行バッファ0～3には常に書き込みスコアボード120と読み出しスコアボード121の出力信号とレジスタファイル107からの演算終了信号124が放送回路125を介して通達される。この通達により、オペランドレジスタへの書き込みが終了し、かつ先行命令によるターゲットレジスタの読み出しが完了した命令が実行可能となる。

【0031】命令選択回路123は命令実行バッファ0～3の中で実行可能となった命令を取り出し、オペランドレジスタの番号をレジスタファイル読み出し回路126へ、命令アドレス、ターゲットレジスタ番号、演算器番号、拡張データを命令発行回路127へ送出する。レジスタファイル読み出し回路126ではオペランドレジスタの内容をレジスタファイル107（図1）から読み出し、オペランドデータとして命令発行回路127へ送り出す。

【0032】命令発行回路127は演算器番号で識別される演算器105にターゲットレジスタ番号、オペランドデータ、拡張データを送り出す。書き込みスコアボード120のメンテナンス動作としては、命令デコーダ103から新規に命令実行バッファ0に到着した命令のオペランドレジスタ（RA、RB両方）の番号が書き込みスコアボード120のセット端子（S）に送られ、該当レジスタのカウント値がインクリメントされる。また、実行可能となった命令のオペランドレジスタ（RA、RB両方）の番号はレジスタファイル読み出し回路126から書き込みスコアボード120のリセット（R）端子に送られ、該当レジスタのカウント値が0になる。読み出しスコアボード121はレジスタ更新をする命令が発行されたときに命令発行回路127からターゲットレジ

スタ番号の通知をセット端子（S）に受け、更新中を示すフラグ1を立てる。このフラグは演算終了信号124によってリセットされる。先に述べた翻訳命令の概念から実行可能となった命令の命令アドレスが実行命令アドレス301として、演算器番号とターゲットレジスタ番号とが転送先物理資源番号302として、オペランドレジスタ番号と拡張データが転送元物理資源番号303として命令翻訳回路204へ送られる。以上の命令スケジューリング回路の動作は、仮に命令デコーダ103の動作にパイプライン1ステージ分の時間がかかるすると、少なくとも2ステージ以上の時間を要する処理となる。

【0033】（4）プロセッサ200

（4a）プログラム例

以下の説明では図5に示したプログラム例を用いる。図5のプログラム例は5種の命令から構成されており、1fd命令501は浮動小数点数値を6番レジスタに格納されている数値とディスプレースメントである8で示されるメモリ番地からロードし、浮動小数点レジスタ1に格納する。ai命令502は6番レジスタの値に4を加算し、再び6番レジスタの値とする。fm命令503は浮動小数点レジスタ1番と2番に格納された数値の積を再び浮動小数点レジスタ1番に格納し、fcmp命令504は浮動小数点レジスタ1番の数値が0番の数値よりも大きい小さいかを比較し、その比較結果を条件レジスタ6番（CR6）に格納する。最後のbc命令505は分岐命令であり、fcmp命令504の結果、浮動小数点レジスタ1番の数値が0番のレジスタの数値よりも小さいとき、ラベル__L10に分岐する。

【0034】この命令列の参照、更新するレジスタと命令の関係をグラフ理論的に図示したものが図6である。この図は命令実行過程を概念的に表すものであり、本実施の形態の装置内の機構と直接対応するものではない。図6からわかるように、グラフ節点601、602、603で示すレジスタは先行命令で更新された後、ただちに後続の命令で参照されており、もし先行命令が更新するレジスタと値を必要としている演算器の両方へ同時に実行結果を転送可能であると、全体の実行時間を短縮できる。このような実行制御方法はフォワーディングと呼ばれ、命令スケジューラ104が実行前の命令列を解釈することによって、転送路を決定する。つまり、スーパーカラープロセッサ100で図5の命令列を実行すると、実行履歴は図7に示したグラフに対応することになる。

【0035】上記のフォワーディングにより、演算結果は最短経路で転送するようにスケジューリングされており、図7の上方および下方には図5の命令列に対する入力値を格納したレジスタ番号（レジスタ名）と出力結果が格納されたレジスタ番号（レジスタ名）が並ぶ。このグラフで命令種を示す節点を演算器と読みかえると、グ

ラフの枝はレジスタと演算器または演算器と演算器のデータ転送を表していることになる。

【0036】(4b) 分岐ブロックの生成

本実施の形態での翻訳命令は、この枝が表すデータ転送の集合であり、枝の横に枝を識別するために付した丸数字が翻訳命令の最小単位である。また、図7において波線で囲んだブロックが示すように図5に示した一群の命令列は、入力値となるブロック670、演算内容となるブロック671、演算結果となるブロック672に分類することが可能である。このことはブロック671を一つのマクロ命令として考えたときに、ブロック670がその実行に対して初期化条件に、ブロック672が終了条件に対応していると考えられる。以上の概念に基づき、命令翻訳回路204では次の手順に従って翻訳命令を作成する。

【0037】入力信号は先に説明した実行命令アドレス301、転送先物理資源番号302、転送元物理資源番号303であり、出力は図11に示した翻訳命令3種類、すなわち初期化命令701、翻訳命令本体702、終了命令703である。また、翻訳命令は一組の初期化命令701と終了命令703と、それらに挟まれた1つまたは複数の翻訳命令本体702で1単位を成す。最初に、翻訳命令生成回路204は1命令実行毎に送られてくる入力信号を次の基準で分岐ブロックに分割する。この分岐ブロックに含まれる複数の命令が翻訳命令の1単位に対応する。

【0038】翻訳命令生成回路204は転送先物理資源番号302を観測し、その転送先が分岐命令処理ユニットであるとき分岐命令が実行されることを認識する。分岐ブロックは分岐命令と分岐命令に挟まれる命令列であり、図5に示した例のように、非分岐命令から始まり、分岐命令で終了する。ただし、ここでは命令実行結果に基づく動的な命令実行順序であるので、例えばアセンブラ言語として図5の命令501の一つ上が分岐命令であるか否かは無関係である。つまり、プログラム中ある分岐命令によって実行アドレスが変わり、命令501にジャンプした場合、命令501から命令505までが一つの分岐ブロックを形成する。また、仮に分岐命令505の条件が成立せず命令501に分岐しなくても、分岐命令505の次に実行される命令から新しい分岐ブロックが開始する。したがって、この定義からプログラム中のある命令が複数の分岐ブロックに属する場合もあり得る。さらにプログラム全体は複数の分岐ブロックにて形成されているので、実行されたプログラム全体が順次翻訳命令に変換されると考えられる。

【0039】(4c) 翻訳命令

翻訳命令生成の機能は二つの機能に大別される。一つは翻訳命令本体702の生成である。図11に示すように、翻訳命令本体702のフィールドはデータ転送の転送先と転送元から構成されており、スーパー scaler

ロッセサ100から出力される転送先物理資源番号302を、転送先フィールド704に転送元物理資源番号303を格納すればよい。翻訳命令本体702の命令長は有限であり、その命令長を越える場合には、新しい翻訳命令本体702を別途生成する。大別した機能の2番目は初期化命令701および終了命令703の生成である。命令翻訳回路204には、後に説明するように、図12に示したスコアボード751と意味的に等価なものが備えられている。スコアボードはスーパー scaler プロセッサ100に内蔵されている個々のレジスタについて、そのレジスタに書き込み動作を実施した演算器の番号と読み出し動作を実施した演算器番号が記録される。なお、分岐ブロックが変わった際には、スコアボード全体が0クリアされる。この番号の記録はスーパー scaler プロセッサ100の命令実行順序に従うため、分岐ブロックが終了した時点では最近に読み出し、あるいは書き込みを実施した演算器の番号が記録されている。

【0040】本実施の形態では翻訳命令を実行するプロセッサがスーパー scaler プロセッサ100とは独立して存在し、レジスタとしては一時レジスタ206を用いる。そのために、一時レジスタ206の初期化が必要である。翻訳命令生成回路204はスコアボード751の中で読み出しはされるが、書き込みが行われていないレジスタを抽出し、初期化命令701を生成する。つまり、上記の条件を満たすレジスタは図7のブロック670のように、着目分岐ブロック外で値が設定されているレジスタである。初期化命令701の転送先フィールドはプロセッサ200の内部に設けられた一時レジスタの番号であり、転送元はレジスタ10207の物理資源番号である。レジスタ10207はスーパー scaler プロセッサ100のレジスタファイル107と一時レジスタ206間のデータ転送機能を有する。なお、初期化命令701の命令開始アドレスには分岐ブロックの先頭命令のアドレスが格納される。

【0041】終了命令703も同様にスコアボード751を参照して生成される。終了命令703は分岐ブロックが終了した際の結果を一時レジスタ206からレジスタファイル107に格納するために存在する。終了命令703の転送先フィールドはレジスタ10207の物理資源番号であり、転送先フィールドは分岐ブロックが終了した時点でスコアボード751に書き込み実績が残っているレジスタ番号である。以上のようにして生成された翻訳命令は分岐ブロック終了と同時に、翻訳命令メモリ201内の書き込み回路214へ転送される。以上が翻訳命令の生成の概略説明である。

【0042】(4d) 命令翻訳回路204

図8に示すように、命令翻訳回路204は、スコアボード751a、751bと初期化命令バッファ130、翻訳命令本体バッファ131、終了命令バッファ132等から構成される。スコアボード751aと751bは、

それぞれ図12に示したスコアボードのRead部とWrite部に記載された情報を保持するスコアボードであり、それぞれこれらのRead部とWrite部に保持された演算器番号を保持する。分岐検出回路133は演算器番号が分岐命令処理ユニットであるとき、リセット信号134と翻訳命令送出要求信号133aを生成する。これにより、先に述べた分岐ブロックが認識できる。リセット信号134を受けたスコアボード751a、751bは、それぞれの記憶内容を全て0クリアする。

【0043】翻訳命令本体の生成は命令スケジューラ104から実行命令アドレス301、転送先物理資源番号302、転送元物理資源番号303が送られる毎に実施される。つまり、図12を用いて説明した概念のとおり、転送元の物理リソースは、スコアボード751bに演算器番号が格納されていれば、その演算器であるし、演算器番号がスコアボード751bに格納されていなければオペランドレジスタが転送元となる。その選択は選択回路135が実施する。また、転送先はターゲットレジスタと演算器のいずれかになる。したがって、2オペランド1ターゲットの命令からは最大3組の転送元と転送先が生成され、それらは翻訳命令バッファ131に格納する。初期化命令と終了命令はリセット信号134に同期して生成される。リセット信号134は0検出回路136aと136bに入力され、それぞれのスコアボード内の格納値が0であるレジスタ番号を出力する。スコアボード751aの0検出回路136aの出力は、終了命令の転送先を指定し、終了命令バッファ132へ格納される。スコアボード751bの0検出回路136bの出力は、条件判定回路137においてスコアボード751aの出力結果と合成される。条件判定回路137の動作内容は、0検出回路136bから出力された書き込み実績の無いレジスタ番号のうち、読み出し実績のあるレジスタ番号を選別することである。条件判定回路137から出力されるレジスタ番号は初期化、すなわちレジスタファイル107の内容を一時レジスタ206へ転送する必要の番号である。プロセッサ200の初期化命令はレジスタ10207を通してレジスタファイル107の内容が転送されるので、転送元がレジスタ10207、転送先が一時レジスタ206となる。この初期化命令は初期化命令バッファ130へ格納される。最後に翻訳命令送出回路138は初期化命令、翻訳命令本体、終了命令をこの順で結合し、翻訳命令メモリ201へ出力する。

【0044】(4e) 翻訳命令メモリ201

図9を参照するに、翻訳命令メモリ201は、書き込み回路214と、翻訳命令記憶領域211と、読み出し回路213とからなる。なお、図には翻訳命令デコーダ202の内部構造も併せて示す。翻訳命令記憶領域211は、MOSトランジスタからなる複数のメモリセル23

3および234と、付随する制御回路(A)235と、制御回路(R)236およびその他の回路とから構成されるメモリアレイである。翻訳命令記憶領域211はより多くのメモリセルが繰り返し存在し、全体としてのセルアレイを構成しているが、それらのメモリセルは簡単化のために図示していない。翻訳命令記憶領域211は、翻訳命令に含まれる命令開始アドレスを格納する複数のメモリセル(M1)233と、転送元となる物理資源番号を格納するための複数のメモリセル(M)234とから構成されている。それぞれのメモリセルは1ビットのセルで、それらの情報を保持するには翻訳命令記憶領域211に設けられた複数ビット分のメモリセルが使用されるが、以下では、簡単化のため必要なビット数分全てのメモリセル233、234は図示しない。また便宜上、水平方向に並ぶメモリセル群をライン、垂直方向に並ぶメモリセル群をカラムと呼ぶ。

【0045】書き込み回路214は、命令翻訳回路204から線219を介して供給される翻訳命令がまだこの翻訳命令メモリ201に記憶されていないときには、この翻訳命令を翻訳命令記憶領域211に格納する。その際、その翻訳命令が、図11に示した初期化命令701であるときには、その命令701は、翻訳命令記憶領域211内のいずれか一つのラインに記憶される。この命令内の各転送元は、転送先となり得る複数の物理資源に一つ一つに対応し、その対応する物理資源に対して定められた水平方向の記憶位置に記憶される。すなわち、その翻訳命令701中の複数の転送先の各々に対応する水平方向の位置に、その転送先と対をなす転送元を記憶する。このために、書き込み回路214では、カラムデコーダ270が翻訳命令中の各転送先をデコードし、翻訳命令記憶領域211のx方向位置を算出する。書き込み増幅器271が、算出された位置にその転送先と対をなす転送元データを書き込む。命令翻訳回路204から順次与えられる複数の翻訳命令は、翻訳命令記憶領域211内のまだ命令が記憶されていないカラムに順次書き込まれる。このような書き込みは、書き込み制御回路212の制御下で行われる。翻訳命令によっては同じ転送先を複数含む場合がある。このような翻訳命令を翻訳命令記憶領域211に格納するときには、転送元情報を複数のラインに分けて翻訳命令記憶領域211に記憶する必要がある。その結果、それらの翻訳命令では、図13と異なって、転送元が稠密に詰まっていない状態として記憶される。

【0046】命令翻訳回路204から与えられた命令が初期化命令702のときには、転送先がデコードされ、翻訳命令702Aとして翻訳命令記憶領域211へ格納される。命令翻訳回路204から与えられたこの翻訳命令が終了命令703であるときには、その命令が指定する最後の転送元を翻訳命令記憶領域211へ書き込んだラインのメモリセル(M1)233には全て1が書き込

10

20

30

40

50

まれる。なお、翻訳回路204から与えられた命令がこの翻訳メモリ201に記録されているか否かは、後に説明するように、ヒット信号242により書き込み制御回路212に伝えられる。もし、この翻訳命令がヒットしなかったときにはこのヒット信号242がLowになる。

【0047】各メモリセル(M1)233は、内容の一致検索機能を持ち、命令フェッチ回路102が生成する命令アドレス300とそのメモリセル内に保持しているデータの一致照合を行う。ある、ラインの命令開始アドレス、つまりメモリセル(M1)233の内容が全て一致した場合には、信号線230の電位がLowとなり、制御回路(A)235が読み出しライン選択信号216をHighに引き上げる。読み出しライン選択信号216がHighになると該当するラインのメモリセルは、メモリセル(M1)233も、メモリセル(M)234も保持していた論理値をデータ出力線218へ出力する。この読み出し動作に関してカラム方向の選択は行わないため、1ライン分のデータが読み出し回路213に到達する。つまり、翻訳メモリ201は命令開始アドレスをタグ情報とした一種のCAM(Contents Associative Memory)として機能する。

【0048】最も基本的なメモリセル(M)234は、図10(b)に示すように、トランジスタT1とトランジスタT2のゲート間容量に電荷を蓄積する。書き込みライン選択信号215がHighのとき、トランジスタT1が開き、データ入力線217の値がこのゲート間容量に記憶される。読み出しライン選択信号216がHighになると、トランジスタT3が開き、記憶していた情報が負論理でデータ出力線218に現れる。データ出力線218は、カラム毎に設けられたブリッジ用トランジスタ238(図9)によって、メモリセルの読み出し前にHighレベルへ引き上げられる。

【0049】メモリセル(M1)233は、図10(a)に示すように、トランジスタT4、T5、T6はそれぞれメモリセル(M)234のトランジスタT1、T2、T3に対応し、同等な役割を持つ。セル内の2つのインバータはそれぞれデータ入力線217と命令アドレス300の反転信号を生成するために存在するが、これらをセル内に記載したのは図面の記載の簡単化のためであり、本来は、カラム毎にこれらの一組のインバータが存在すればよい。トランジスタT7とT4およびトランジスタT8とT9の結線にはそれぞれ書き込んだデータの正論理値と負論理値の両方が記憶され、トランジスタT10のゲートには、このセルに記憶されているデータと命令アドレス300の排他的NORの論理値が最終的に生成される。つまり、照会情報として与えられた命令開始アドレスがこのセルに記憶されている情報と一致した場合、信号線230はこのメモリセル(M1)23

3内で導通状態となる。

【0050】制御回路(A)235は、図10(d)に示す回路からなる。クロックの立ち上がりより前は、トランジスタT14が信号線230を電源電位にブリッジする。先に述べたように、メモリセル(M1)233内のトランジスタT10が全て導通しているときは、クロックの立ち上がり以降、信号線230はLowとなる。読み出し回路213が与える強制的読み出し要求(Instruction request)237

は、後述する動作の時以外にはLowレベルである。このため、メモリセル(M1)233で検出された一致は、ゲートg1を経て読み出しライン選択信号216をHighに設定することになる。このようにして、翻訳命令記憶領域211からは、命令アドレス300に一致したラインのデータがデータ出力線218に現れる。

【0051】制御回路(R)236は、リフレッシュ制御240、ラインデコーダ239と協調してメモリセルのリフレッシュを実施する回路で、図10(c)に示す構造を有する。リフレッシュ時にはリフレッシュ制御240がリフレッシュ信号232をHighレベルにした状態で通常の読み出し動作を実行する。つまり、ラインデコーダ239がリフレッシュするラインの読み出しライン選択信号216をHighにする。制御回路(R)236のトランジスタT11とT12のゲート間にデータが保持される。次にラインデコーダ239は書き込みライン選択信号215をHighレベルに上げ、リフレッシュ制御240がリフレッシュ信号232をLowにすることで、制御回路(R)236に保持されていたデータはブリッジインバータを経て元のラインに書き戻される。また、ORゲート241は全読み出しライン選択信号の論理的ORをとることにより、先に述べた命令開始アドレスの検索におけるヒット信号242を生成する。

【0052】こうして、命令翻訳回路204から初めて翻訳メモリ201に供給された命令に対してもこのヒットチェックが行われ、その命令がヒットしなかったときには先に記載したように、その命令がこの翻訳メモリ201に書き込まれる。さらに、すでにその命令が翻訳メモリ201に書き込まれた後に、再度実行されたときには、その命令が属する命令ブロックの先頭の命令の命令アドレスに対するヒットチェックの結果、その先頭の命令がヒットする。その結果、この命令から始まる一連の命令に対する複数の翻訳命令が順次翻訳メモリ201から読み出されることになる。

【0053】読み出し回路213には、転送先となり得る複数の物理資源に対応して、複数のFIFOキュー260が設けられ、各FIFOキュー260は、センスアンプを介して対応する物理資源に対して設けられたデータ出力線218に接続されている。翻訳命令記憶領域211から転送先データ出力線218に読み出されたアナ

ログ信号は、このセンスアンプで論理信号に確定された後、FIFOキュー260へ書き込まれる。この書き込みは、ラッチ261にオール1が読み出されるまで、すなわち、終了命令が読み出されるまで繰り返される。このとき、ラインアドレスのインクリメントはFIFOキュー260からラインデコーダ239へ伝えられる。

【0054】(4f) 翻訳命令の実行態様

図9において、翻訳命令デコーダ202は、翻訳命令の各転送元フィールドをデコードする複数の部分デコーダ202Aと命令待機ユニット802からの転送要求であるinstruction request 808から命令読み出し回路213内部のFIFOキュー260への転送要求信号202cを生成するORゲート202Bとからなり、各部分デコーダ202Aに対して、全ての物理資源、すなわち、複数の演算器205、複数の一時レジスタ206、レジスタ10207の全てに対応する複数のinstruction信号801が設けられている。先に述べたように、翻訳命令メモリ201内部では命令が要求する転送先物理資源の番号は、翻訳命令記憶領域211内の位置情報として記憶されているので、翻訳命令デコーダ202は、転送元物理資源のみをデコードすればよい。つまり、各部分デコーダは、読み出された翻訳命令の中の対応する転送元フィールドをデコードし、その転送元物理資源に対応する一つのinstruction信号を選択して起動する。このデコード結果は、その部分デコーダが対応する転送先となる物理資源、すなわち、演算器205、一時レジスタ206またはレジスタ10207のいずれか一つが待つべきデータがどの転送元の物理資源で生成されるのかを示している。各instruction信号801は1ビットのデータ線から構成される。

【0055】図14において、転送回路203は、全ての物理資源、すなわち、複数の演算器205、複数の一時レジスタ206およびレジスタ10207の各々の各入力端に対応して、その物理資源に対応する部分デコーダの複数のinstruction信号801と、それぞれのinstruction信号801が対応する物理資源の出力線との交点に命令待機ユニット802が設けられている。各物理資源の各入力端に対応して設けられた複数の命令待機ユニット802の出力は、ワイア

40 オアされて、その入力端に供給される。

【0056】命令待機ユニット802の内部構成を図15に示す。instruction信号801は先に述べた翻訳命令デコーダ202で生成された信号であり、着目している命令待機ユニット802の関係する転送元と転送先との間にデータ転送が必要であることを意味する信号である。

【0057】instruction request 808は命令待機ユニット802から翻訳命令デコーダ202へのデータ転送要求であり、翻訳命令デコーダ

02内のゲートでOR論理がとられ、読み出し回路213内のFIFOキューから新たな転送元データが取り出される。data-in信号803は所定のデータ幅、例えば32ビット幅のデータ線であり、転送元からデータを伝送する。data-in信号803に新しいデータが到着したことはdata-in-valid信号805が有効になることで示される。同様に、data-out信号804は転送先へのデータ転送線であり、data-in信号803と同様に32ビット等の幅が用意される。data-out信号804の有効性はdata-out-valid信号806によって示される。命令待機ユニット802の基本的な役割はinstruction信号801が有効、すなわちデータ転送が必要な命令が存在するときに、data-in信号803の内容をdata-out信号804に出力することである。転送先の演算器やレジスタはそれぞれの機能を実現するために必要なデータが揃った時点で動作を開始する。

20 【0058】より具体的には上記の制御が転送制御回路807で実現される。転送制御回路807の回路と機能は図16に示した回路図およびタイムチャートの通りである。つまり、ラッチ8000とゲート8002の組およびラッチ8001とゲート8003の組はそれぞれinstruction信号801とdata-in-valid信号805がHighになった状態をdata-out-valid信号806がHighになるまで保持する。ゲート8004は上記2つの信号がともにHighになった状態を検出し、ラッチ8005の状態を1サイクルの間反転させる。このラッチ8005の出力はdata-out-valid信号806、instruction request 808として出力される。instruction信号801が有効である場合、この命令待機ユニット802は転送元からのデータを待っている状態であり、次にdata-in信号803にデータが到着すると、転送先の演算器やレジスタの格納動作を起動する意味でdata-out信号804から同一データを出力する。図14に戻り、data-in信号803およびdata-in-valid信号805は同一水平位置にある命令待機ユニット802に全て接続され、電気信号は一斉に伝達するよう接続されている。したがって、先に述べたように命令が転送元と転送先に接続される結線の交差する位置で実行を待機する動作が実現できる。

【0059】次に翻訳命令の実行形態を視覚的に把握する目的で、図17から図20に図5で例示したプログラムの実行例を示す。図17から図20において丸数字は図7の丸数字に対応しており、翻訳命令のデータ転送をあらわしている。また、丸数字の0は翻訳命令の初期化命令に、丸数字の12は終了命令に対応している。図17の上方に囲んだ領域900は翻訳命令メモリ201に

格納された翻訳命令を表し、下方の領域901は図14において命令待機ユニット802が設けられていた領域に対応する。領域901の下に並べられた図1における演算器205～レジスタ10207に対応しており、説明の便宜上実行するプログラムと同じ演算名とレジスタ名で対応関係を示している。なおレジスタ10207は図17から図20においては1つだけ例示しているが、これは図面の都合上であり、データ転送が必要なレジスタの数だけレジスタ10も存在する。また図17中の丸数字12下方にある黒丸は、翻訳命令の初期化命令を起

動するために存在し、初期化命令毎、かつレジスタ10207対応に存在する。また、本発明の原理から一時レジスタ206の物理資源番号はレジスタファイル107内のレジスタ番号と一致している必要がある。初期化命令によって一時レジスタにレジスタデータの初期値を転送する際、一時レジスタ206の物理資源番号も同時に付与され、以降の命令実行ではその物理資源番号が使用される。

【0060】翻訳命令はinstruction信号801が空の場合にはただちに領域900から領域901の命令待機ユニット802へ転送される。すなわち、翻訳命令の実行は図17の状態からただちに図18の状態へ移行する。図17に存在した黒丸のデータ転送命令はレジスタ10207を起動し、レジスタファイル107から所望の(図7からわかるように、本プログラム例ではfp0、r6、fp2の初期内容)データを読み出し、レジスタ10の出力結果とする。図15の命令待機レジスタ10の出力線902上には丸数字0の命令が待機しており(図18)、レジスタ10を介してレジスタファイル107の内容が信号線803に出力されると、丸数字0の直下に位置づけられたそれぞれの演算器へデータを転送し、演算を起動する。ここで信号線803は命令待機ユニット802のdata-in信号803と同一であり、各演算器、一時レジスタの出力内容は命令待機ユニット802へ一斉に送られる。

【0061】以降の処理はいわゆる玉突き状態で進行する。例えば、丸数字0によってr6(レジスタ6番)の内容が読み出されると図18に示すように、その出力線上に待機している丸数字1と丸数字10に対応してai(加算)とlfd(浮動小数点データのロード)を実施する演算器へr6のデータが転送される。fp2(浮動小数点レジスタ2番)、fp0(浮動小数点レジスタ0番)に対応した丸数字0のデータ転送命令も同様であり、結果として図19の状態を得る。図19の状態からは丸数字の11によって12が実行され、加算後のr6の内容がレジスタファイルに書き戻され、図20の状態となる。最終的には丸数字12で示したデータ転送が実施され、図7のブロック672が示すように、r6と条件レジスタCR6、浮動小数点レジスタfp1の値がレジスタファイル107に書き込まれる。

【0062】以上の翻訳命令実行では命令待機ユニットで待機中の命令は真に必要なデータのみを待つために、命令スケジューラ104で実施されていた従来技術のように演算を終了した全データと待機中の全命令との照合等が不要となり、高速な命令待機と実行が実現できる。また、この過程が従来技術より高速であるということは、同じ実行内容の命令列でも、翻訳命令メモリ201に蓄積された命令列の方が、従来の命令デコーダ103、命令スケジューラ104を経由する実行より高速であることを意味する。

【0063】以上に示した実施の形態によれば、命令実行に必要な物理資源の割り当て、決定に関しては最初の命令実行時には従来技術と同様な手段により認識するため、処理時間の改善はない。しかし、2回目以降の同一命令の実行に関しては翻訳命令メモリ201内に格納されているデータ転送先、データ転送元の情報を利用するため、スーパースcalarプロセッサ100内の命令デコーダ103による処理と命令スケジューラ104による処理を省略することができ、その分の高速化が可能となる。実質的效果は、同じ命令列を利用する割合、すなわち翻訳メモリのヒット率に依存する。命令キャッシュなどの研究から知られているように、実行命令の局所性は高く、特に翻訳メモリ201を同一チップ内のDRAM等で実現すると、ヒット率はほぼ100%を実現することが可能である。

【0064】また、本実施の形態では、実行すべき命令が実行に必要なデータをデータが転送されてくる経路上で待つため、リザーベーションステーションのように、明らかに関連性のない命令に対して命令実行可能性のチェックを行うようなことはない。事実、従来技術では上記の冗長な機能のために、実行待ちの命令は自分の必要とするデータを番号で管理しており、演算終了時に告知される番号との照合が必要である。このことから、一つのリザーベーションステーションに待機可能な命令は4命令程度が限度であり、比較にも時間を必要とした。本実施の形態では、転送先、転送元の物理資源情報は転送回路203中の位置情報として利用され、最終的に命令として転送回路203内で待機する命令は実行を起動するための1ビットの情報でしかない。このことから、実行条件の成立の判定は高速であり、構造が簡単になることから多くの命令を実行待機させることができる。また、データの転送先に対応した場所で命令実行を待機するため、存在する演算器数以上の命令を実行しようとすることもないし、それらの調停回路も必要でなくなる。

【0065】<変形例>本発明は、以上の実施例に限定されるものではなく、以下に示す変形例およびその他のいろいろの変形例として、実施可能である。

【0066】(1)実施の形態1ではスーパースcalarプロセッサ100の複数の演算器105、レジスタファイル107とは別にプロセッサ200用に複数の演算器

10

20

30

40

50

205、複数の一時レジスタ206が設けられた。しかし、この後者の演算器とレジスタに代えて、前者に含まれた演算器およびレジスタを使用することも可能である。このためには、実施の形態1の転送回路203を、スーパースカラープロセッサ100内の演算器105とレジスタファイル107内のレジスタの間でデータ転送するように、これらの演算器とレジスタに接続する。このときには、プロセッサ200内の翻訳メモリ201、翻訳命令デコーダ202、転送回路203等の回路を動作させるクロックは、スーパースカラープロセッサ100を動作させるクロックと同じとすることが現実的である。したがって、実施の形態1で述べたような高速な動作は実現できないが、翻訳命令を使用する結果、命令スケジューラ104使用しないで翻訳命令を実行できるので、この点で従来より高速な動作が期待できる。

【0067】(2) 上記の実施形態では図11に示した翻訳命令の転送先を命令翻訳回路204でデコードし、翻訳命令メモリ201内部には図13のように転送元だけの情報を記録していた。これは、図14に示すように、関連するレジスタや演算器と翻訳命令メモリ内のデータを対応づけておいた方が、翻訳命令の取り出しから実行までの時間が短縮されるからである。しかし、この方法は翻訳命令メモリ201の利用効率が必ずしも高くないため、図11に示した翻訳命令の形態をそのまま翻訳命令メモリ201内に格納することも可能である。ただし、この方法では翻訳命令読み出し時に転送先に関わるデコードをするため、本発明の主眼である、命令実行までの処理時間は若干犠牲となる。

【0068】(3) 翻訳命令メモリとしてはSRAMやDRAMといった半導体メモリを想定しているが、たとえば図21に示すように、ハードディスクのような外部記憶装置201Aに格納する形態も可能である。翻訳命令フェッチ回路201Bは読み出し回路213と同様に翻訳命令を蓄積する手段(本変形例では外部記憶装置201A)から翻訳命令の転送元情報を読み出す。レジスタファイル107Aはレジスタファイル107と論理、物理両面で同一のレジスタファイルである。ただし、この場合には上述の実施形態とは異なり、翻訳命令はスカラープロセッサ100の動作と同時に実行されず、別途プロセッサ200上で実行される。

【0069】

【発明の効果】本発明によれば、逐次実行用のプログラムの命令に対してスケジュール処理を施してそれらの命令を並列に実行した後、スケジュール処理の結果をメモ

リに格納するので、再度そのプログラムを実行するときには、そのスケジュール処理を再度実行する必要がなく、そのプログラムをより高速に再実行できる。

【図面の簡単な説明】

【図1】本発明に係る情報処理装置の概略ブロック図。

【図2】図1の装置における命令実行のフローチャート。

【図3】図1の装置に使用されるスカラープロセッサ100の概略構成図。

10 【図4】図1の装置に用いる命令のフォーマットを示す図。

【図5】図1の装置で実行されるプログラムの例を示す図。

【図6】図5のプログラムのグラフ表現を示す図。

【図7】図5のプログラムのデータ転送を表すグラフ表現を示す図。

【図8】図1の装置に使用される命令翻訳回路の概略ブロック図。

20 【図9】図1の装置に使用する翻訳メモリの概略ブロック図。

【図10】図9の装置に使用される複数種のメモリセルと複数の制御回路の回路図。

【図11】図8の装置により生成される翻訳命令のフォーマットを示す図。

【図12】図8の装置内に含まれた二つのスコアボードに保持された内容と同じ内容を保持する等価なスコアボードを示す図。

【図13】図9の装置に記憶された翻訳命令のフォーマットを示す図。

30 【図14】図1の装置に含まれた翻訳命令デコーダと転送回路の概略構成図。

【図15】図14の装置に使用される命令待機ユニットのブロック図。

【図16】図15の装置の回路図とタイムチャート。

【図17】図1の装置における翻訳命令の第1の実行例を示す図。

【図18】図1の装置における翻訳命令の第2の実行例を示す図。

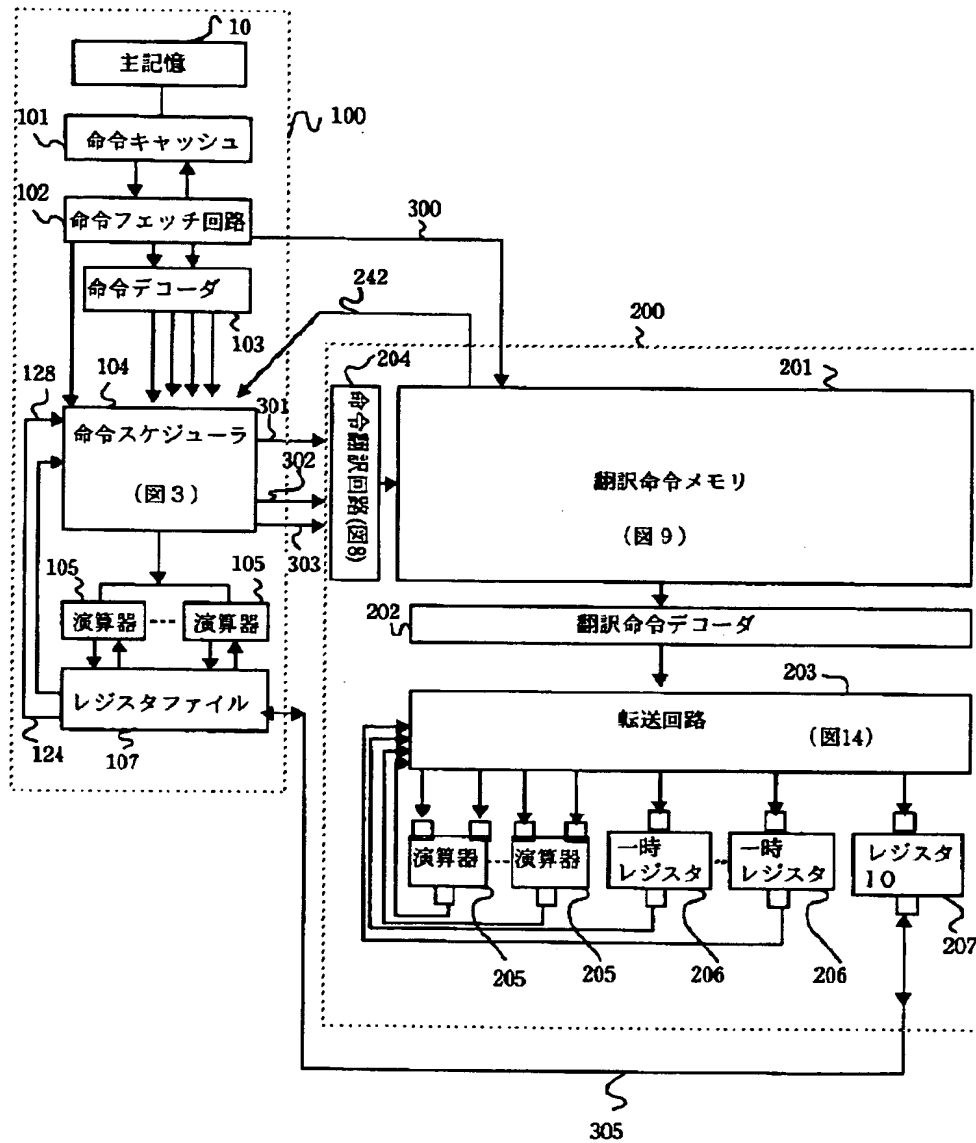
40 【図19】図1の装置における翻訳命令の第3の実行例を示す図。

【図20】図1の装置における翻訳命令の第4の実行例を示す図。

【図21】本発明に係る他の情報処理装置の概略ブロック図。

【圖 1】

图 1



【図4】

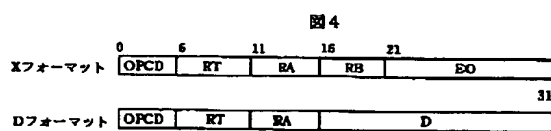
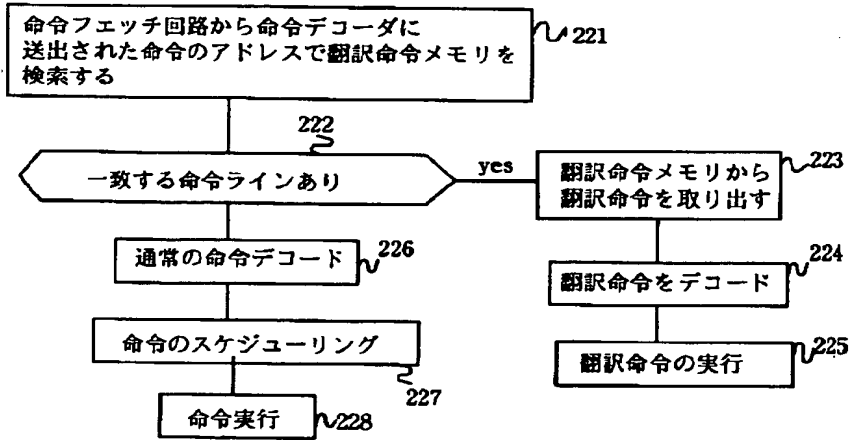


圖 4

【図2】

図2



【図5】

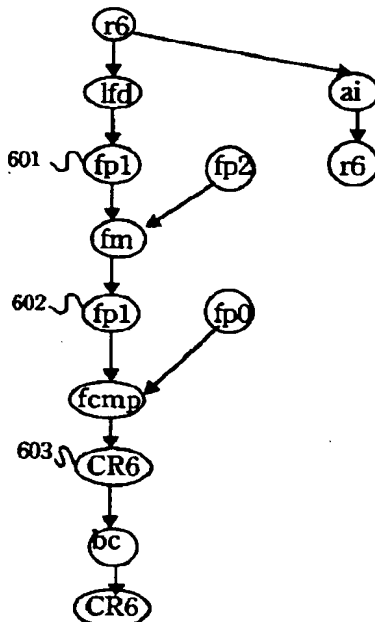
図5

```

_L10: lfd    fp1 8(r6) ~ 501
      ai    r6, r6, 4 ~ 502
      fm    fp1, fp2, fp1 ~ 503
      fcmp  cr6, fp0, fp1 ~ 504
      bc    IF, CR6_LT, _L10 ~ 505
  
```

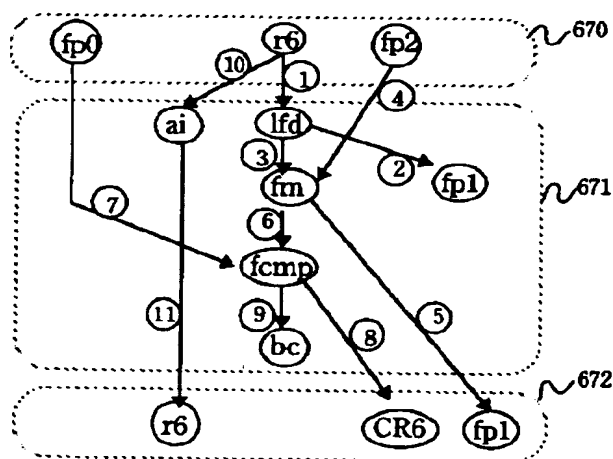
【図6】

図6



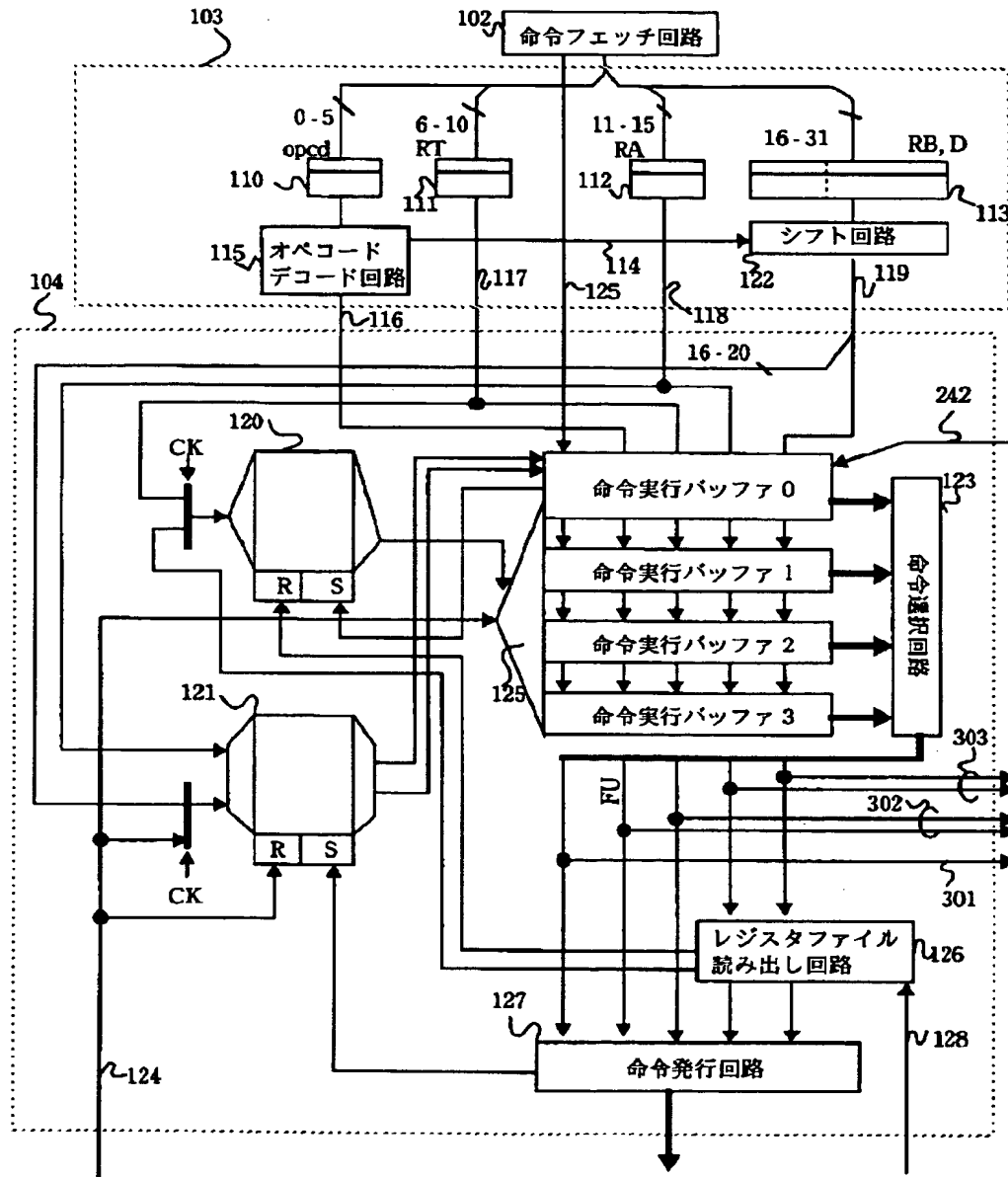
【図7】

図7



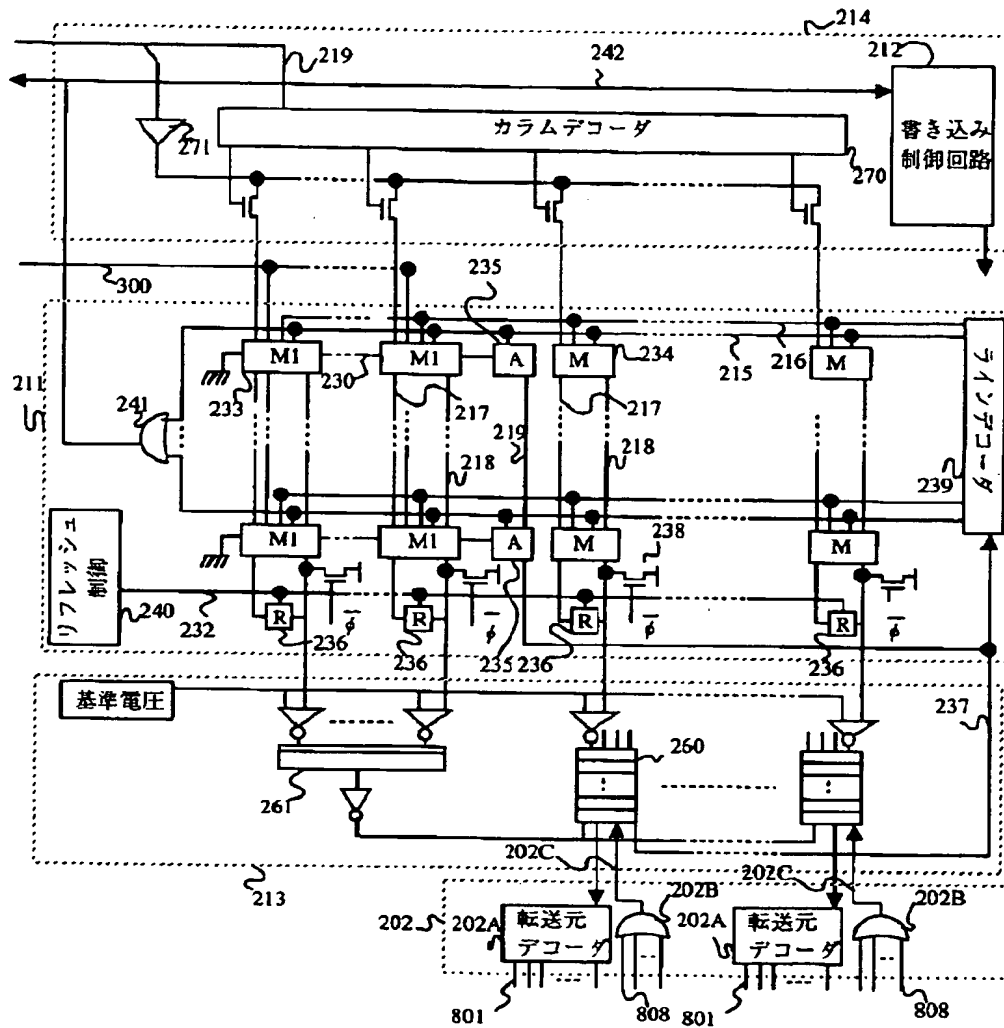
【図3】

図3



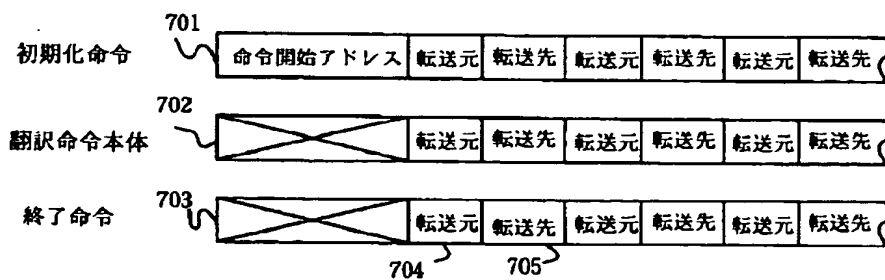
【図9】

図9



【図11】

図11



【図12】

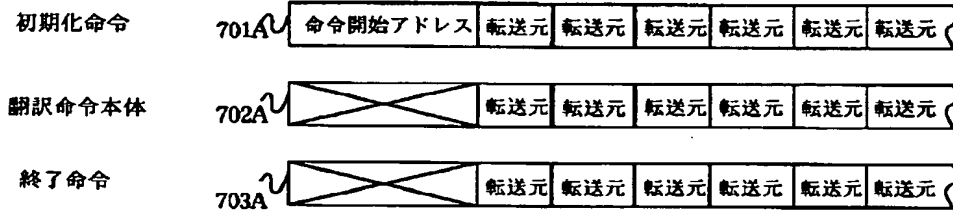
図12

751

	r6		fp1		fp2		fp0		CR6	
	read	write	read	write	read	write	read	write	read	write
lfd fp1 8(r6)	lfd	0	0	lfd	0	0	0	0	0	0
ai r6, r6, 4	ai	ai	0	0	0	0	0	0	0	0
fm fp1, fp2, fp1	0	0	fm	fm	fm	0	0	0	0	0
fcmp 6, fp0, fp1	0	0	fcmp	0	0	0	fcmp	0	0	fcmp
bc IF, CR6_LT, _L10	0	0	0	0	0	0	0	0	bc	0

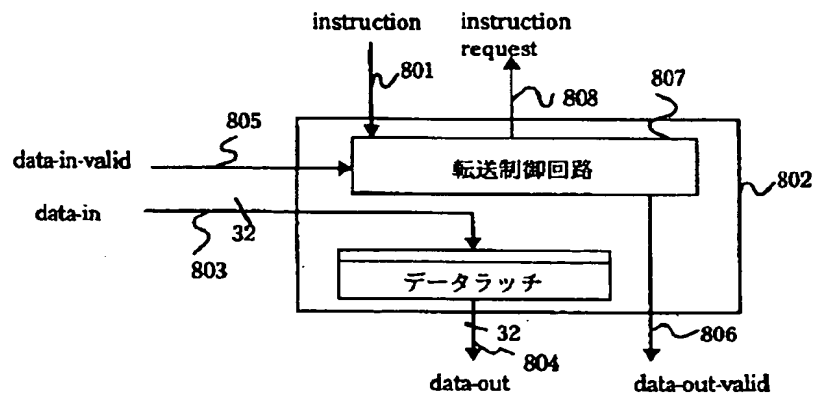
【図13】

図13



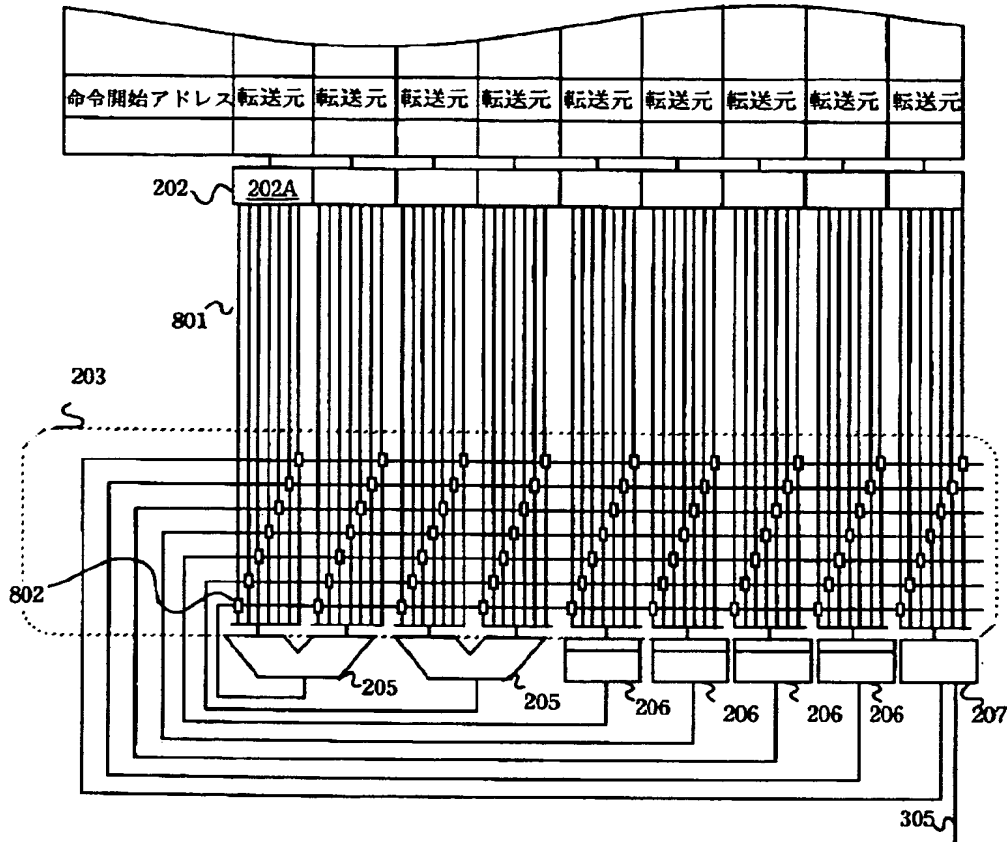
【図15】

図15



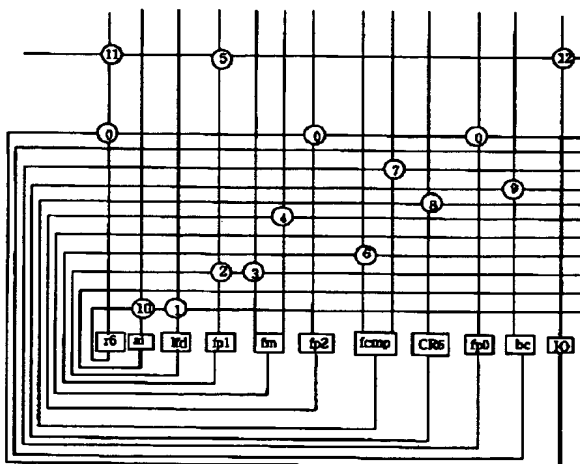
【図14】

図14



【図18】

図18



【図20】

図20

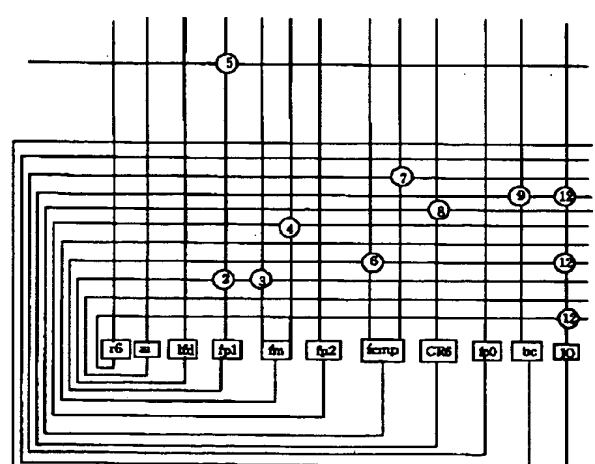
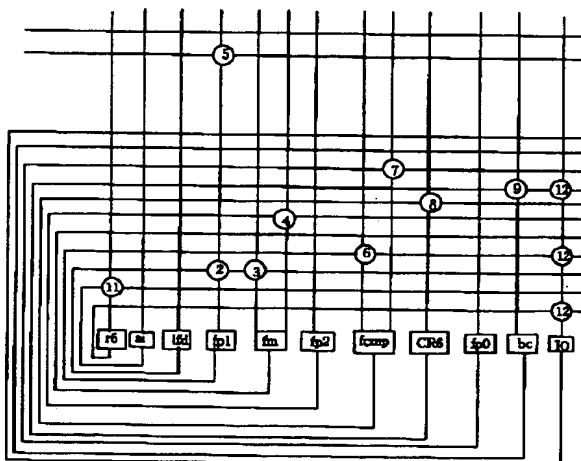


图16



圖19



【図21】

図21

